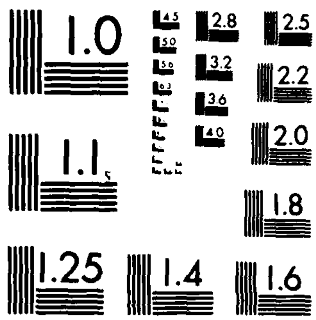
AD-A128 836 A FORTRAN SUBROUTINE TO EVALUATE SPHERICAL BESSEL FUNCTIONS OF THE FIRST..(U) NAVAL RESEARCH LAB WASHINGTON DC J P MASON 23 MAY 83 NRL-8705 1/1

UNCLASSIFIED F/G 9/2 NL

END
DATE
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

May 23, 1983

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Report 8705	2. GOVT ACCESSION NO. AD-A128836	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A FORTRAN SUBROUTINE TO EVALUATE SPHERICAL BESSEL FUNCTIONS OF THE FIRST, SECOND, AND THIRD KINDS FOR COMPLEX ARGUMENTS		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) J. P. Mason		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61153N; RR011-08-41 51-0382-0-3
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		12. REPORT DATE May 23, 1983 13. NUMBER OF PAGES 16
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Bessel functions
Spherical Bessel functions

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A FORTRAN subroutine, CSPJYD, has been written for the VAX-750 that will generate a table of spherical Bessel functions of the first, second, and third kinds for a given complex argument and a given range of integer orders.

DD FORM 1473, 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

CONTENTS

A

A FORTRAN SUBROUTINE TO EVALUATE SPHERICAL BESSEL FUNCTIONS OF THE FIRST, SECOND, AND THIRD KINDS FOR COMPLEX ARGUMENTS

1. INTRODUCTION

To determine analytically the acoustic scattering from absorbing spheres, it is necessary to evaluate spherical Bessel functions of the first, second, and third kinds for complex arguments, covering a range of integer orders from zero to a value approximately equal to the absolute value of the argument. A double-precision (G-format) FORTRAN subroutine, CSPJYD, has been written for the VAX-750 that will generate tables of $j_n(z)$ and $y_n(z)$ and/or $h_n^{(k)}(z)$ for a given value of z and a range of n from zero to a given maximum n.

2. METHODS OF COMPUTATION

For an absolute value of the argument less than or equal to 0.5, $j_n(z)$ and $y_n(z)$ are each generated by an alternating series; they are then used to generate the $h_n^{(k)}(z)$ values. For arguments of absolute value greater than 0.5, when the absolute value of the coefficient of the imaginary part of the argument is less than 5.0, $j_n(z)$ and $y_n(z)$ are always calculated, and conditional on the subroutine call, only then is $h_n^{(k)}(z)$ derived. However, when the absolute value of the coefficient of the imaginary part of the argument is greater than or equal to 5.0, $j_n(z)$ and $h_n^{(k)}(z)$ are always calculated, and $y_n(z)$ only if requested in the subroutine call. The values for $y_n(z)$, or $h_n^{(k)}(z)$, will not be returned to the calling program unless the user so requests, even though those functions were calculated. See Sec. 4, Instructions for Use, for further clarification.

In the following equations, $z = u + iv$ was used in place of the more common notation $z = x + iy$, to prevent confusing the coefficient of the imaginary part of the argument with the spherical Bessel function of the second kind, $y_n(z)$.

a. if $|z| \leqslant 0.5$,

$$j_n(z) = \frac{z^n}{1 \cdot 3 \cdot 5 \cdots (2n+1)} \left\{1 - \frac{(z^2/2)}{1!(2n+3)} + \frac{(z^2/2)^2}{2!(2n+3)\,(2n+5)} - \cdots\right\}$$

$$y_n(z) = \frac{-1 \cdot 3 \cdot 5 \cdots (2n-1)}{z^{n+1}} \left\{1 - \frac{(z^2/2)}{1!(1-2n)} + \frac{(z^2/2)^2}{2!(1-2n)\,(3-2n)} - \cdots\right\}$$

$$\left.\begin{aligned} h_n^{(1)}(z) &= j_n(z) + iy_n(z) \\ h_n^{(2)}(z) &= j_n(z) - iy_n(z) \end{aligned}\right\} \begin{aligned} &h_n^{(1)}(z) \text{ is calculated if } v \geqslant 0 \\ &h_n^{(2)}(z) \text{ is calculated if } v < 0 \end{aligned}$$

b. if $|z| > 0.5$ and $|v| < 5.0$,

$$\left.\begin{aligned} j_0(z) &= \frac{\sin z}{z} \\ j_1(z) &= \frac{\sin z}{z^2} - \frac{\cos z}{z} \\ y_0(z) &= \frac{-\cos z}{z} \\ y_1(z) &= \frac{-\cos z}{z^2} - \frac{\sin z}{z} \end{aligned}\right|$$

where:

$$\sin z = \sin u(e^v + e^{-v})/2 + i\,[\cos u\,(e^v - e^{-v})/2] \text{ and}$$

$$\cos z = \cos u(e^v + e^{-v})/2 - i[\sin u(e^v - e^{-v})/2]$$

Manuscript approved February 15, 1983.

$j_n(z) = (2n+3)z^{-1}\, j_{n+1}(z) - j_{n+2}(z)$ (backward recursion)

$y_n(z) = (2n-1)z^{-1}\, y_{n-1}(z) - y_{n-2}(z)$ (forward recursion)

$$\left.\begin{aligned} h_n^{(1)}(z) &= j_n(z) + iy_n(z) \\ h_n^{(2)}(z) &= j_n(z) - iy_n(z) \end{aligned}\right\} \begin{aligned} &h_n^{(1)}(z) \text{ is calculated if } v \geqslant 0 \\ &h_n^{(2)}(z) \text{ is calculated if } v < 0 \end{aligned}$$

c. if $|z| > 0.5$ and $|v| \geqslant 5.0$,

$$\left.\begin{aligned} j_0(z) &= \frac{\sin z}{z} \\ j_1(z) &= \frac{\sin z}{z^2} - \frac{\cos z}{z} \end{aligned}\right\} \text{where } \sin z \text{ and } \cos z \text{ are defined as in } b.$$

$$\left.\begin{aligned} h_0^{(1)}(z) &= e^{-v}(\sin u - i\cos u)/z \\ h_1^{(1)}(z) &= e^{-v}(\sin u - i\cos u)/z^2 - e^{-v}(\cos u + i\sin u)/z \\ h_0^{(2)}(z) &= e^{v}(\sin u + i\cos u)/z \\ h_1^{(2)}(z) &= e^{v}(\sin u + i\cos u)/z^2 - e^{v}(\cos u - i\sin u)/z \end{aligned}\right\} \begin{aligned} &h_0^{(1)}(z) \text{ and } h_1^{(1)}(z) \text{ are calculated if } v \geqslant 0 \\ &h_0^{(2)}(z) \text{ and } h_1^{(2)}(z) \text{ are calculated if } v < 0 \end{aligned}$$

$j_n(z) = (2n+3)z^{-1}\, j_{n+1}(z) - j_{n+2}(z)$ (backward recursion)

$h_n^{(k)}(z) = (2n-1)z^{-1}\, h_{n-1}^{(k)}(z) - h_{n-2}^{(k)}(z)$ (forward recursion)

$y_n(z) = ij_n(z) - ih_n^{(1)}(z)$ or

$y_n(z) = -\, ij_n(z) + ih_n^{(2)}(z)$

3. VERIFICATION

Because published tables often are limited in range or lack precision, the accuracy of the derived function values was checked with at least one of the following Wronskian relationships:

a. $j_{n+1}(z)\, y_n(z) - j_n(z)\, y_{n+1}(z) = 1/z^2$

b. $[j_n(z)\, h_{n+1}^{(1)}(z) - j_{n+1}(z)\, h_n^{(1)}(z)]\, i = 1/z^2$

c. $-[j_n(z)\, h_{n+1}^{(2)}(z) - j_{n+1}(z)\, h_n^{(2)}(z)]i = 1/z^2$

The table below lists some of the arguments and orders for which runs were made, the degree of accuracy in the resultant Bessel functions,* and their respective Wronskians. Note that these Wronskians were calculated as functions of $j_{n-1}(z)$, $j_n(z)$, $h_{n-1}^{(k)}(z)$, and $h_n^{(k)}(z)$, in every case.

Argument			max n	Accuracy of Results at max n (places)	Wronskian Agreement at max n (figures; places)
0.01	−	0.001i	20	10	14;10
1.0	−	100.0i	100	10	14;19
100.0	±	0.5i	100	10	13;17
100.0	−	100.0i	100	10	16;20
1000.0	−	10.0i	100	10	11;18
1000.0	−	100.0i	100	10	15;21
0.0	±	0.4i	10	(a)	16;15
0.0	±	0.6i	10	(a)	15;14
0.0	±	5.1i	10	(a)	15;16

(a) No comparison was made.

*The accuracy was determined by comparing these results with a 10-place table of spherical Bessel functions of the first and second kinds.

4. INSTRUCTIONS FOR USE

The subroutine call is:

CALL CSPJYD (AR, AI, N, SJR, SJI, SYR, SYI, SHR, SHI, NAK)

input:
- AR = the real part of the argument z.
- AI = the imaginary part of the argument z.
- N = the maximum order.
- NAK = 2, if $j_n(z)$ and $y_n(z)$ are to be calculated.
- NAK = 3, if $j_n(z)$ and $h_n^{(k)}(z)$ are to be calculated.
- NAK = 5, if all three are to be calculated.

output:
- SJR = a table, from $n = 0$ to $n = N$, of the real part of $j_n(z)$.
- SJI = a table, from $n = 0$ to $n = N$, of the imaginary part of $j_n(z)$.
- SYR = a table, from $n = 0$ to $n = N$, of the real part of $y_n(z)$ if NAK = 2 or 5.
- SYR = a table, from $n = 0$ to $n = N$, of the real part of $h_n^{(k)}(z)$ if NAK = 3.
- SYI = a table, from $n = 0$ to $n = N$, of the imaginary part of $y_n(z)$ if NAK = 2 or 5.
- SYI = a table, from $n = 0$ to $n = N$, of the imaginary part of $h_n^{(k)}(z)$ if NAK = 3.
- SHR = a table, from $n = 0$ to $n = N$, of the real part of $h_n^{(k)}(z)$ if NAK = 5.
- SHI = a table, from $n = 0$ to $n = N$, of the imaginary part of $h_n^{(k)}(z)$ if NAK = 5.

If NAK = 2 or 3, for SHR and SHI use dummy parameters, which do not have to be dimensioned. All parameters except N and NAK must be REAL*8. The routine calculates $h_n^{(1)}(u + iv)$ for positive v and $h_n^{(2)}(u + iv)$ for negative v. SJR and SJI should be dimensioned by at least $(u^2 + v^2)^{1/2} + 30$ or $N + 30$, whichever is the larger. SYR and SYI should be dimensioned by at least $N + 1$. If NAK = 2 or 3, SHR and SHI require no dimensioning, but if NAK = 5, they also should be dimensioned by at least $N + 1$.

Subroutine CSPJYD calls two other subroutines, DVDD and MLTD; they perform complex division and multiplication, respectively.

5. PORTABILITY

Generally speaking, CSPJYD can be adapted for use on many other computers. It works with integers and double-precision real values only, so there is no problem in using it on a computer, such as the PDP-11, which permits no higher precision for its complex numbers than COMPLEX*8. The checks for overflow, divide by zero, and underflow, which are located in subroutines CSPJYD and DVDD, would have to be changed for non-DEC machines, but this should not prove difficult.

6. LISINGS AND OUTPUT[1]

Following are source listings of subroutines CSPJYD, DVDD, and MLTD; a listing of the test program TSPHBF; and output from two sample runs of program TSPHBF. One note here: if both $y_n(z)$ and $h_n^{(k)}(z)$ are printed, as in examples 3 and 4, it is $h_n^{(1)}(z)$ that is used with $j_n(z)$ to determine the Wronskian check.

[1] M. Abramowitz and I.A. Stegun, eds., 1965, *Handbook of Mathematical Functions*, U.S. Department of Commerce, National Bureau of Standards, Washington, DC 20402.

```
      SUBROUTINE CSPJYD(AR,AI,N,SJR,SJI,SYR,SYI,SHR,SHI,NAK)
C     AS OF 30 AUGUST 1982
C     DOUBLE-PRECISION SPHERICAL BESSEL FUNCTIONS FOR COMPLEX ARGUMENTS
C     WRITTEN BY J.P.MASON, ACOUSTICS DIVISION, NRL
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION SJR(1),SJI(1),SYR(1),SYI(1),SHR(1),SHI(1)
      LOGICAL LT,LF
      DATA LT/.TRUE./,LF/.FALSE./
      CALL ERRSET(72,LF,LF,LF,LT,)
      CALL ERRSET(73,LF,LF,LF,LT,)
      CALL ERRSET(74,LF,LF,LF,LT,)
C     WRITE(5,104)
104   FORMAT(' SPHERICAL BESSEL FUNCTIONS FOR COMPLEX ARG')
      ZERO=0.0D0
      ONE=1.0D0
      TWO=2.0D0
      THREE=3.0D0
      IZ=0
      DR=AR*AR-AI*AI
      DI=TWO*AR*AI
      CC=TWO
      EPS=1.0D-16
      WUNR=ONE
      WUNI=ZERO
      CALL DVDD(WUNR,WUNI,DR,DI,T1,T2)
      SRARG=DSQRT(AR*AR+AI*AI)
      IF(SRARG.GT.0.5D0)GO TO 29
      NP=N+1
      CALL MLTD(AR,AI,AR,AI,ZR,ZI)
      ZR=ZR/TWO
      ZI=ZI/TWO
      FDNM=THREE
      HDN=ONE
      HDNM=ONE
      HDNI=ZERO
      DO 14 I=1,NP
      NN=I-1
      EN=NN
C     CALCULATE Z**N/(1X3X5...(2N+1)) FOR J
      IF(NN-1)2,6,3
6     FNR=AR/THREE
      FNI=AI/THREE
      GO TO 5
2     FNR=ONE
      FNI=ZERO
      GO TO 5
3     CALL MLTD(FNR,FNI,AR,AI,FNR,FNI)
      FDNM=FDNM+TWO
      FNR=FNR/FDNM
      FNI=FNI/FDNM
5     ANSR=ONE
      ANSI=ZERO
      PANSR=ONE
      PANSI=ZERO
      TRM=-ONE
      TIM=ZERO
```

```
      AB=ONE
      BA=THREE
7     GNU=AB*(TWO*EN+BA)
      ZRS=-ZR/GNU
      ZIS=-ZI/GNU
      CALL MLTD(TRM,TIM,ZRS,ZIS,TRM,TIM)
      ANSR=ANSR-TRM
      ANSI=ANSI-TIM
      IF(ANSR.EQ.ZERO)GO TO 15
      IF(ANSI.EQ.ZERO)GO TO 16
      IF(DABS((PANSR-ANSR)/ANSR).LE.EPS.AND.DABS((PANSI-ANSI)/ANSI)
     1 .LE.EPS)GO TO 8
      GO TO 17
15    IF(DABS((PANSI-ANSI)/ANSI).LE.EPS)GO TO 8
      GO TO 17
16    IF(DABS((PANSR-ANSR)/ANSR).LE.EPS)GO TO 8
17    PANSR=ANSR
      PANSI=ANSI
      AB=AB+ONE
      BA=BA+TWO
      GO TO 7
8     CALL MLTD(FNR,FNI,ANSR,ANSI,SJR(I),SJI(I))
C     CALCULATE (-1X3X5...(2N-1))/Z**(N+1) FOR Y
      IF(NN-1)4,10,9
4     GDR=-ONE
      GDI=ZERO
      CALL DVDD(GDR,GDI,AR,AI,HR,HI)
      GO TO 11
10    HDR=AR
      HDI=AI
9     CALL MLTD(HDR,HDI,AR,AI,HDR,HDI)
      HDNM=HDNM*HDN
      HDN=HDN+TWO
      CALL DVDD(HDNM,HDNI,HDR,HDI,HR,HI)
      HR=-HR
      HI=-HI
11    ALSR=ONE
      ALSI=ZERO
      PALSR=ONE
      PALSI=ZERO
      TRN=-ONE
      TIN=ZERO
      AC=ONE
      CA=ONE
12    HNU=AC*(CA-TWO*EN)
      XRS=-ZR/HNU
      XIS=-ZI/HNU
      CALL MLTD(TRN,TIN,XRS,XIS,TRN,TIN)
      ALSR=ALSR-TRN
      ALSI=ALSI-TIN
      IF(ALSR.EQ.ZERO)GO TO 18
      IF(ALSI.EQ.ZERO)GO TO 19
      IF(DABS((PALSR-ALSR)/ALSR).LE.EPS.AND.DABS((PALSI-ALSI)/ALSI)
     1 .LE.EPS)GO TO 13
      GO TO 20
18    IF(DABS((PALSI-ALSI)/ALSI).LE.EPS)GO TO 13
```

```
      GO TO 20
19    IF(DABS((PALSR-ALSR)/ALSR).LE.EPS)GO TO 13
20    PALSR=ALSR
      PALSI=ALSI
      AC=AC+ONE
      CA=CA+TWO
      GO TO 12
13    CALL MLTD(HR,HI,ALSR,ALSI,SYR(I),SYI(I))
C     WRITE(5,200)I,SYR(I),SYI(I)
C200  FORMAT(I5,2(1X,D23.16))
C     IF NAK=2, PUT Y'S IN SYR AND SYI.
C     IF NAK=3, STORE Y'S; PUT H'S INTO SYR AND SYI.
C     IF NAK=5, PUT Y'S INTO SYR AND SYI; PUT H'S INTO SHR AND SHI.
      IF(NAK.EQ.2)GO TO 14
      IF(NAK.EQ.5)GO TO 50
      YRR=SYR(I)
      YII=SYI(I)
      IF(AI.LT.ZERO)GO TO 51
      SYR(I)=SJR(I)-YII
      SYI(I)=SJI(I)+YRR
      GO TO 14
51    SYR(I)=SJR(I)+YII
      SYI(I)=SJI(I)-YRR
      GO TO 14
50    IF(AI.LT.ZERO)GO TO 48
      SHR(I)=SJR(I)-SYI(I)
      SHI(I)=SJI(I)+SYR(I)
      GO TO 14
48    SHR(I)=SJR(I)+SYI(I)
      SHI(I)=SJI(I)-SYR(I)
14    CONTINUE
      RETURN
29    DSN=DSIN(AR)
      DCS=DCOS(AR)
      EXYL=DEXP(AI)
      EXYS=DEXP(-AI)
      XSN=AR*DSN
      XCO=AR*DCS
      YSN=AI*DSN
      YCO=AI*DCS
      ZXY=AR*AR+AI*AI
      TZXY=TWO*ZXY
      SJZRL=(XSN+YCO)/TZXY
      SJZRS=(XSN-YCO)/TZXY
      SJZIL=(XCO-YSN)/TZXY
      SJZIS=(-XCO-YSN)/TZXY
      SYZRL=EXYL*(-SJZIL)
      SYZRS=EXYS*SJZIS
      SYZIL=EXYL*SJZRL
      SYZIS=EXYS*(-SJZRS)
      SJZRL=EXYL*SJZRL
      SJZRS=EXYS*SJZRS
      SJZIL=EXYL*SJZIL
      SJZIS=EXYS*SJZIS
      SJR(1)=SJZRL+SJZRS
      SJI(1)=SJZIL+SJZIS
```

```
      SJR(2)=((AR*SJZRL+AI*SJZIL)/ZXY+SYZRL)+
     1        ((AR*SJZRS+AI*SJZIS)/ZXY+SYZRS)
      SJI(2)=((-AI*SJZRL+AR*SJZIL)/ZXY+SYZIL)+
     1        ((-AI*SJZRS+AR*SJZIS)/ZXY+SYZIS)
      NHO=0
      IF(DABS(AI).LT.5.0D0)GO TO 43
C     CALCULATE HANKEL FUNCTIONS AND THEN THE NEUMANN FUNCTIONS.
      NHO=1
      YEX=DEXP(-DABS(AI))
      ANUR=YEX*DSN
      ANUI=YEX*DCS
      IF(AI.GE.ZERO)ANUI=-ANUI
      CALL DVDD(ANUR,ANUI,AR,AI,HRZ,HIZ)
      CALL MLTD(AR,AI,AR,AI,ZSR,ZSI)
      CALL DVDD(ANUR,ANUI,ZSR,ZSI,HRW,HIW)
      IF(AI)38,39,39
38    ANUR=-ANUR
      GO TO 40
39    ANUI=-ANUI
40    CALL DVDD(ANUI,ANUR,AR,AI,HOA,HOB)
C     IF NAK=2, STORE H'S; PUT Y'S INTO SYR AND SYI.
C     IF NAK=3, PUT H'S INTO SYR AND SYI
C     IF NAK=5, PUT Y'S INTO SYR AND SYI; PUT H'S INTO SHR AND SHI.
      IF(NAK.LT.5)GO TO 54
      SHR(1)=HRZ
      SHI(1)=HIZ
      SHR(2)=HRW-HOA
      SHI(2)=HIW-HOB
      GO TO 55
54    IF(NAK.EQ.2)GO TO 56
      SYR(1)=HRZ
      SYI(1)=HIZ
      SYR(2)=HRW-HOA
      SYI(2)=HIW-HOB
      GO TO 36
56    HRW=HRW-HOA
      HIW=HIW-HOB
      SYR(1)=-SJI(1)+HIZ
      SYI(1)=SJR(1)-HRZ
      SYR(2)=-SJI(2)+HIW
      SYI(2)=SJR(2)-HRW
      GO TO 57
55    SYR(1)=-SJI(1)+SHI(1)
      SYI(1)=SJR(1)-SHR(1)
      SYR(2)=-SJI(2)+SHI(2)
      SYI(2)=SJR(2)-SHR(2)
57    IF(AI.GE.ZERO)GO TO 36
      SYR(1)=-SYR(1)
      SYI(1)=-SYI(1)
      SYR(2)=-SYR(2)
      SYI(2)=-SYI(2)
      GO TO 36
C     CALCULATE NEUMANN FUNCTIONS AND THEN THE HANKEL FUNCTIONS.
43    SYR(1)=SYZRL+SYZRS
      SYI(1)=SYZIL+SYZIS
      SYR(2)=((AR*SYZRL+AI*SYZIL)/ZXY-SJZRL)+
```

```
     1          ((AR*SYZRS+AI*SYZIS)/ZXY-SJZRS)
        SYI(2)=((-AI*SYZRL+AR*SYZIL)/ZXY-SJZIL)+
     1          ((-AI*SYZRS+AR*SYZIS)/ZXY-SJZIS)
C       IF NAK=2, PUT Y'S INTO SYR AND SYI.
C       IF NAK=3, STORE Y'S; PUT H'S INTO SYR AND SYI.
C       IF NAK=5, PUT Y'S INTO SYR AND SYI; PUT H'S INTO SHR AND SHI.
42      IF(NAK.EQ.2)GO TO 36
        IF(NAK.EQ.5)GO TO 52
        YRZ=SYR(1)
        YIZ=SYI(1)
        YRW=SYR(2)
        YIW=SYI(2)
        IF(AI.LT.ZERO)GO TO 53
        SYR(1)=SJR(1)-YIZ
        SYI(1)=SJI(1)+YRZ
        SYR(2)=SJR(2)-YIW
        SYI(2)=SJI(2)+YRW
        GO TO 36
53      SYR(1)=SJR(1)+YIZ
        SYI(1)=SJI(1)-YRZ
        SYR(2)=SJR(2)+YIW
        SYI(2)=SJI(2)-YRW
        GO TO 36
52      IF(AI.LT.ZERO)GO TO 41
        SHR(1)=SJR(1)-SYI(1)
        SHI(1)=SJI(1)+SYR(1)
        SHR(2)=SJR(2)-SYI(2)
        SHI(2)=SJI(2)+SYR(2)
        GO TO 36
41      SHR(1)=SJR(1)+SYI(1)
        SHI(1)=SJI(1)-SYR(1)
        SHR(2)=SJR(2)+SYI(2)
        SHI(2)=SJI(2)-SYR(2)
36      IF(N.LE.1)RETURN
C       THE J'S, Y'S, AND H'S FOR N=0 AND N=1 HAVE BEEN GENERATED.
        M=N+1
C       FIND REMAINING J'S.
        NN=SRARG+30
        IF((N+30).GT.NN)NN=N+30
        GDR=SJR(2)
        GDI=SJI(2)
30      SJR(NN)=ZERO
        SJI(NN)=ZERO
        SJR(NN-1)=1.0D-20
        SJI(NN-1)=1.0D-20
        NM=NN-2
        DO 31 K=2,NM
        KK=NN-K
        CALL DVDD(SJR(KK+1),SJI(KK+1),S[illegible]R(KK),SJI(KK))
        CALL ERRSET(72,LT,LF,LF,LF,)
        CALL ERRSNS
        SJR(KK)=(CC*KK+ONE)*SJR(KK)-SJR(KK+2)
        CALL ERRSNS(NUM,,,)
        IF(NUM.EQ.72)GO TO 24
        CALL ERRSNS
        SJI(KK)=(CC*KK+ONE)*SJI(KK)-SJI(KK+2)
```

```
      CALL ERRSET(72,LF,LF,LF,LT,)
      CALL ERRSNS(NUM,,,)
      IF(NUM.EQ.72)GO TO 24
31    CONTINUE
      CALL DVDD(GDR,GDI,SJR(2),SJI(2),RAR,RAI)
C     IF THERE WAS AN UNDERFLOW IN THE DVDD SUBROUTINE AND EITHER RAR
C       OR RAI WAS MADE EQUAL TO ZERO, NN SHOULD BE REDUCED.
      IF(RAR.NE.ZERO.AND.RAI.NE.ZERO)GO TO 67
      IF(DABS(SJR(2)).LT.DABS(SJI(2)))GO TO 68
      IF(RAR.NE.ZERO)GO TO 69
      IF(GDR.EQ.ZERO.AND.SJI(2).EQ.ZERO)GO TO 69
      GO TO 24
69    IF(RAI.NE.ZERO)GO TO 67
      IF(GDI.EQ.ZERO.AND.SJI(2).EQ.ZERO)GO TO 67
      GO TO 24
68    IF(RAR.NE.ZERO)GO TO 70
      IF(GDI.EQ.ZERO.AND.SJR(2).EQ.ZERO)GO TO 70
      GO TO 24
70    IF(RAI.NE.ZERO)GO TO 67
      IF(GDR.EQ.ZERO.AND.SJR(2).EQ.ZERO)GO TO 67
      GO TO 24
67    DO 32 K=3,M
      TR=SJR(K)
      TI=SJI(K)
32    CALL MLTD(TR,TI,RAR,RAI,SJR(K),SJI(K))
      SJR(2)=GDR
      SJI(2)=GDI
C     FIND REMAINING Y'S AND H'S.
      IF(NHO.EQ.1)GO TO 44
C     IF NAK=2, PUT Y'S INTO SYR AND SYI.
C     IF NAK=3, STORE Y'S; PUT H'S INTO SYR AND SYI.
C     IF NAK=5, PUT Y'S INTO SYR AND SYI; PUT H'S INTO SHR AND SHI.
      IF(NAK.EQ.3)GO TO 66
22    DO 23 K=3,M
      CALL DVDD(SYR(K-1),SYI(K-1),AR,AI,SYR(K),SYI(K))
      SYR(K)=(CC*K-THREE)*SYR(K)-SYR(K-2)
      SYI(K)=(CC*K-THREE)*SYI(K)-SYI(K-2)
      IF(NAK.EQ.2)GO TO 23
      IF(AI.LT.ZERO)GO TO 45
47    SHR(K)=SJR(K)-SYI(K)
      SHI(K)=SJI(K)+SYR(K)
      GO TO 23
45    SHR(K)=SJR(K)+SYI(K)
      SHI(K)=SJI(K)-SYR(K)
23    CONTINUE
      RETURN
66    DO 60 K=3,M
      CALL DVDD(YRW,YIW,AR,AI,YRT,YIT)
      YRT=(CC*K-THREE)*YRT-YRZ
      YIT=(CC*K-THREE)*YIT-YIZ
      IF(AI.LT.ZERO)GO TO 58
      SYR(K)=SJR(K)-YIT
      SYI(K)=SJI(K)+YRT
      GO TO 59
58    SYR(K)=SJR(K)+YIT
      SYI(K)=SJI(K)-YRT
```

```
59      YRZ=YRW
        YIZ=YIW
        YRW=YRT
        YIW=YIT
60      CONTINUE
        RETURN
C       IF NAK=2, STORE H'S; PUT Y'S INTO SYR AND SYI.
C       IF NAK=3, PUT H'S INTO SYR AND SYI.
C       IF NAK=5, PUT Y'S INTO SYR AND SYI; PUT H'S INTO SHR AND SHI.
44      IF(NAK.NE.5)GO TO 61
        DO 46 K=3,M
        CALL DVDD(SHR(K-1),SHI(K-1),AR,AI,SHR(K),SHI(K))
        SHR(K)=(CC*K-THREE)*SHR(K)-SHR(K-2)
        SHI(K)=(CC*K-THREE)*SHI(K)-SHI(K-2)
        SYR(K)=-SJI(K)+SHI(K)
        SYI(K)=SJR(K)-SHR(K)
        IF(AI.GE.ZERO)GO TO 46
        SYR(K)=-SYR(K)
        SYI(K)=-SYI(K)
46      CONTINUE
        RETURN
61      IF(NAK.EQ.3)GO TO 62
        DO 63 K=3,M
        CALL DVDD(HRW,HIW,AR,AI,HRT,HIT)
        HRT=(CC*K-THREE)*HRT-HRZ
        HIT=(CC*K-THREE)*HIT-HIZ
        SYR(K)=-SJI(K)+HIT
        SYI(K)=SJR(K)-HRT
        IF(AI.GE.ZERO)GO TO 64
        SYR(K)=-SYR(K)
        SYI(K)=-SYI(K)
64      HRZ=HRW
        HIZ=HIW
        HRW=HRT
        HIW=HIT
63      CONTINUE
        RETURN
62      DO 65 K=3,M
        CALL DVDD(SYR(K-1),SYI(K-1),AR,AI,SYR(K),SYI(K))
        SYR(K)=(CC*K-THREE)*SYR(K)-SYR(K-2)
        SYI(K)=(CC*K-THREE)*SYI(K)-SYI(K-2)
65      CONTINUE
        RETURN
24      NN=NN-1
        WRITE(5,26)NN
26      FORMAT (1X,' NN REDUCED TO ',I6)
        IZ=IZ+1
        IF(IZ.GT.25)RETURN
        GO TO 30
        END
$
```

```
      SUBROUTINE DVDD(XA,YA,XB,YB,XC,YC)
C     AS OF 11 JANUARY 1983
C       WRITTEN BY JANET P. MASON
      IMPLICIT REAL*8 (A-H, -Z)
      LOGICAL LT,LF
      DATA LT/.TRUE./,LF/.FALSE./
      ZERO=0.0D0
      IF(XB.NE.ZERO.OR.YB.NE.ZERO)GO TO 3
      WRITE(5,100)
      WRITE(6,100)
  100 FORMAT (' BOTH REAL AND IMAGINARY PARTS OF DENOMINATOR ARE ZERO')
      RETURN
    3 CALL ERRSET(72,LT,LF,LF,LF,)
      CALL ERRSET(73,LT,LF,LF,LF,)
      CALL ERRSET(74,LT,LF,LF,LF,)
      DENOM=XB*XB+YB*YB
      IF(DENOM.EQ.ZERO)GO TO 1
      XX=(XA*XB+YA*YB)/DENOM
      IF(XX.EQ.ZERO)GO TO 1
      YC=(YA*XB-XA*YB)/DENOM
      IF(YC.EQ.ZERO)GO TO 1
      XC=XX
      RETURN
    1 CALL ERRSET(72,LF,LF,LF,LT,)
      CALL ERRSET(73,LF,LF,LF,LT,)
      CALL ERRSET(74,LF,LF,LF,LT,)
      IF(DABS(XB).LT.DABS(YB))GO TO 2
    8 DC=YB/XB
      AC=XA/XB
      BC=YA/XB
      CALL ERRSET(74,LT,LF,LF,LF,)
      DENOM=1.0D0+DC*DC
C       IF DC*DC UNDERFLOWS, DENOM WILL EQUAL 1.0D0
      XC=(AC+BC*DC)/DENOM
      YC=(BC-AC*DC)/DENOM
      CALL ERRSET(74,LF,LF,LF,LT,)
      RETURN
    2 AD=XA/YB
      CD=XB/YB
      BD=YA/YB
      CALL ERRSET(74,LT,LF,LF,LF,)
      DENOM=1.0D0+CD*CD
C       IF CD*CD UNDERFLOWS, DENOM WILL EQUAL 1.0D0
      XC=(BD+AD*CD)/DENOM
      YC=(-AD+BD*CD)/DENOM
      CALL ERRSET(74,LF,LF,LF,LT,)
      RETURN
      END
$
      SUBROUTINE MLTD(XA,YA,XB,YB,XC,YC)
C     AS OF 31 JULY 1978
C       WRITTEN BY JANET P. MASON
      IMPLICIT REAL*8 (A-H,O-Z)
      XX=XA*XB-YA*YB
      YC=XA*YB+YA*XB
      XC=XX
      RETURN
      END
$
```

```
      PROGRAM TSPHBF
C     AS OF 11 JANUARY 1983
C     WRITTEN BY JANET P. MASON
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER NA=2,NB=1202
      DIMENSION X(NA),Y(NA),MAX(NA),NAF(3)
      DIMENSION A(NB),B(NB),C(NB),D(NB),E(NB),F(NB)
      DATA X/1000.0D0,1000.0D0/
      DATA Y/600.0D0,600.0D0/
      DATA MAX/4,1167/
      DATA NAF/2,3,5/
      ZERO=0.0D0
      ONE=1.0D0
      EXD=1.0D+153
      DO 5 L=3,3
      DO 1 I=1,NA
      AR=ZERO
      AI=ZERO
      NMAX=MAX(I)+1
      IF(NAF(L).NE.5)GO TO 8
      CALL CSPJYD(X(I),Y(I),NMAX,A,B,C,D,E,F,NAF(L))
      GO TO 9
8     CALL CSPJYD(X(I),Y(I),NMAX,A,B,C,D,G,H,NAF(L))
9     CALL MLTD(X(I),Y(I),X(I),Y(I),ZSR,ZSI)
      CALL DVDD(ONE,ZERO,ZSR,ZSI,ZSR,ZSI)
      WRITE(5,6)X(I),Y(I),ZSR,ZSI
6     FORMAT(7X,'Z = ',2(1X,D23.16),/,' 1/(Z*Z) = ',2(1X,D23.16)
     1        ,//,29X,'REAL PART',14X,'IMAGINARY PART',/)
      NNMAX=NMAX+1
      DO 4 J=1,NNMAX
      NN=J-2
      IF(J.EQ.1)GO TO 4
      IF(NAF(L)-3)7,10,12
7     IF(DABS(A(J)).GT.EXD.OR.DABS(B(J)).GT.EXD.
     1   OR.DABS(C(J-1)).GT.EXD.OR.DABS(D(J-1)).GT.EXD.
     2   OR.DABS(A(J-1)).GT.EXD.OR.DABS(B(J-1)).GT.EXD.
     3   OR.DABS(C(J)).GT.EXD.OR.DABS(D(J)).GT.EXD)GO TO 13
      CALL MLTD(A(J),B(J),C(J-1),D(J-1),RRR,RRI)
      CALL MLTD(A(J-1),B(J-1),C(J),D(J),SRR,SRI)
      AR=RRR-SRR
      AI=RRI-SRI
      GO TO 13
12    BR=B(J)*E(J-1)+A(J)*F(J-1)-B(J-1)*E(J)-A(J-1)*F(J)
      BI=-A(J)*E(J-1)+B(J)*F(J-1)+A(J-1)*E(J)-B(J-1)*F(J)
      IF(Y(I).GE.ZERO)GO TO 13
      BR=-BR
      BI=-BI
      GO TO 13
10    AR=B(J)*C(J-1)+A(J)*D(J-1)-B(J-1)*C(J)-A(J-1)*D(J)
      AI=-A(J)*C(J-1)+B(J)*D(J-1)+A(J-1)*C(J)-B(J-1)*D(J)
      IF(Y(I).GE.ZERO)GO TO 13
      AR=-AR
      AI=-AI
13    IF(MAX(I).GT.20.AND.NN.LT.MAX(I)-4)GO TO 4
      IF(NAF(L)-3)14,16,18
14    WRITE(5,15)NN,A(J-1),B(J-1),C(J-1),D(J-1),AR,AI
```

```
15      FORMAT(' N = ',I4,2X,'SPHJ(Z) = ',2(1X,D23.16),/,
     1                      11X,'SPHY(Z) = ',2(1X,D23.16),/,
     2                    9X,'WRONSKIAN = ',2(1X,D23.16))
        GO TO 4
16      WRITE(5,17)NN,A(J-1),B(J-1),C(J-1),D(J-1),AR,AI
17      FORMAT(' N = ',I4,2X,'SPHJ(Z) = ',2(1X,D23.16),/,
     1                      11X,'SPHH(Z) = ',2(1X,D23.16),/,
     2                    9X,'WRONSKIAN = ',2(1X,D23.16))
        GO TO 4
18      WRITE(5,19)NN,A(J-1),B(J-1),C(J-1),D(J-1),E(J-1),F(J-1),BR,BI
19      FORMAT(' N = ',I4,2X,'SPHJ(Z) = ',2(1X,D23.16),/,
     1                      11X,'SPHY(Z) = ',2(1X,D23.16),/,
     2                      11X,'SPHH(Z) = ',2(1X,D23.16),/,
     3                    9X,'WRONSKIAN = ',2(1X,D23.16))
4       CONTINUE
        WRITE(5,2)
2       FORMAT(///)
1       CONTINUE
5       CONTINUE
        STOP
        END
$
```

Example 1

```
      Z =  -0.1000000000000000D-02 -0.1000000000000000D-03
1/(Z*Z) =   0.9704930889128516D+06 -0.1960592098813842D+06

                          REAL PART                IMAGINARY PART

N =    0  SPHJ(Z) =   0.9999998350000079D+00 -0.3333333003333344D-07
          SPHY(Z) =   0.9900985099010306D+03 -0.9900995099008657D+02
        WRONSKIAN =   0.9704930889128519D+06 -0.1960592098813842D+06
N =    1  SPHJ(Z) =  -0.3333333010000011D-03 -0.3333332336666725D-04
          SPHY(Z) =  -0.9704935889127281D+06  0.1960592098814092D+06
        WRONSKIAN =   0.9704930889128518D+06 -0.1960592098813842D+06
N =    2  SPHJ(Z) =   0.6599999552333345D-07  0.1333333144761912D-07
          SPHY(Z) =   0.2824417825519075D+10 -0.8706194121960350D+09
        WRONSKIAN =   0.9704930889128513D+06 -0.1960592098813841D+06
N =    3  SPHJ(Z) =  -0.9238094761640223D-11 -0.2847618788354505D-11
          SPHY(Z) =  -0.1355126578346419D+14  0.5708223540316741D+13
        WRONSKIAN =   0.9704930889128516D+06 -0.1960592098813843D+06
```

Example 2

```
      Z =  -0.1000000000000000D-02 -0.1000000000000000D-03
1/(Z*Z) =   0.9704930889128516D+06 -0.1960592098813842D+06

                          REAL PART                IMAGINARY PART

N =    0  SPHJ(Z) =   0.9999998350000079D+00 -0.3333333003333344D-07
          SPHH(Z) =  -0.9800995115508655D+02 -0.9900985099343640D+03
        WRONSKIAN =   0.9704930889128519D+06 -0.1960592098813842D+06
N =    1  SPHJ(Z) =  -0.3333333010000011D-03 -0.3333332336666725D-04
          SPHH(Z) =   0.1960592095480759D+06  0.9704935888793948D+06
        WRONSKIAN =   0.9704930889128518D+06 -0.1960592098813842D+06
N =    2  SPHJ(Z) =   0.6599999552333345D-07  0.1333333144761912D-07
          SPHH(Z) =  -0.8706194121960349D+09 -0.2824417825519075D+10
        WRONSKIAN =   0.9704930889128514D+06 -0.1960592098813841D+06
N =    3  SPHJ(Z) =  -0.9238094761640223D-11 -0.2847618788354505D-11
          SPHH(Z) =   0.5708223540316741D+13  0.1355126578346419D+14
        WRONSKIAN =   0.9704930889128516D+06 -0.1960592098813843D+06
```

Example 3

```
        Z =    0.1000000000000000D+04  0.6000000000000000D+03
 1/(Z*Z) =    0.3460207612456747D-06 -0.6487889273356401D-06

                                 REAL PART                IMAGINARY PART

N =      0  SPHJ(Z) =   0.1615056579356138+258  0.9189988821784188+256
            SPHY(Z) =  -0.9189988821784188+256  0.1615056579356138+258
            SPHH(Z) =   0.9538545182398760-264 -0.2062840276226553-263
          WRONSKIAN =   0.3460207612456747D-06 -0.6487889273356401D-06
N =      1  SPHJ(Z) =  -0.9067180254704272+256  0.1614411627841877+258
            SPHY(Z) =  -0.1614411627841877+258 -0.9067180254704272+256
            SPHH(Z) =  -0.2063048989202653-263 -0.9557921307304426-264
          WRONSKIAN =   0.3460207612456748D-06 -0.6487889273356402D-06
N =      2  SPHJ(Z) =  -0.1613119869413142+258 -0.8821867930011386+256
            SPHY(Z) =   0.8821867930011386+256 -0.1613119869413142+258
            SPHH(Z) =  -0.9596703805949663-264  0.2063462417247416-263
          WRONSKIAN =   0.3460207612456748D-06 -0.6487889273356401D-06
N =      3  SPHJ(Z) =   0.8454661476397939+256 -0.1611177608568539+258
            SPHY(Z) =   0.1611177608568539+258  0.8454661476397939+256
            SPHH(Z) =   0.2064072544606159-263  0.9654953095745764-264
          WRONSKIAN =   0.3460207612456748D-06 -0.6487889273356401D-06
N =      4  SPHJ(Z) =   0.1608579340256789+258  0.7966475353394592+256
            SPHY(Z) =  -0.7966475353394592+256  0.1608579340256789+258
            SPHH(Z) =   0.9732759600894194-264 -0.2064867297777066-263
          WRONSKIAN =   0.3460207612456747D-06 -0.6487889273356401D-06
```

Example 4

```
        Z =    0.1000000000000000D+04  0.6000000000000000D+03
 1/(Z*Z) =    0.3460207612456747D-06 -0.6487889273356401D-06

                                 REAL PART                IMAGINARY PART

N = 1163    SPHJ(Z) =  -0.3760144898599847+107  0.4750248660846224+107
            SPHY(Z) =  -0.4750248660846224+107 -0.3760144898599847+107
            SPHH(Z) =  -0.4608145843562346-113  0.3825673442412044-113
          WRONSKIAN =   0.3460207612456768D-06 -0.6487889273356398D-06
N = 1164    SPHJ(Z) =  -0.3098693455489950+107  0.3133046845640833+106
            SPHY(Z) =  -0.3133046845640833+106 -0.3098693455489950+107
            SPHH(Z) =  -0.8928271566952981-114  0.1161106000839288-112
          WRONSKIAN =   0.3460207612456769D-06 -0.6487889273356398D-06
N = 1165    SPHJ(Z) =  -0.1224447080537097+107 -0.1029806863014308+107
            SPHY(Z) =   0.1029806863014308+107 -0.1224447080537097+107
            SPHH(Z) =   0.1500954349634534-112  0.1697564672546519-112
          WRONSKIAN =   0.3460207612456768D-06 -0.6487889273356397D-06
N = 1166    SPHJ(Z) =  -0.5900803226283226+105 -0.8191635895987654+106
            SPHY(Z) =   0.8191635895987654+106 -0.5900803226283226+105
            SPHH(Z) =   0.4407619877450161-112  0.2049149537982679-113
          WRONSKIAN =   0.3460207612456768D-06 -0.6487889273356397D-06
N = 1167    SPHJ(Z) =   0.2800861011330237+106 -0.3146852038771679+106
            SPHY(Z) =   0.3146852038771679+106  0.2800861011330237+106
            SPHH(Z) =   0.6270970008025791-112 -0.5882652699931355-112
          WRONSKIAN =   0.3460207812456767D-06 -0.6487889273356398D-06
```